AVL2

- root : AVLNODE*
- insert (Word*&, AVLNODE*&) : void
- makeEmpty (AVLNODE*&) : void
- print (ostream&, AVLNODE*) const : void
- returnWord (string&, AVLNODE*) : Word*

+ AVL2 ()
+ AVL2 (const AVL2 &)
+ ~AVL2()
+ get Root() : AVLNode *
+ height (AVLNODE*&) : int
+ insert (Word*&) : void
+ isEmpty () const : bool
+ make Empty () : void
+ return Word (string &) : Word*
+ rotate With Left Child (AVLNODE*&) : void
+ rotate With Right Child (AVLNODE*&) : void
+ double With Left Child (AVLNODE*&) : void
+ double With Right Child (AVLNODE*&) : void
+ print (ostream &) const : void
+ search For (string &, AVLNODE*) : bool

AVLNODE

struct
    element : Word*
    height : int
    ~~AVLNODE*?~~
    left : AVLNODE*
    right : AVLNODE*
    AVLNODE (Word*, AVLNODE*, AVLNODE*, int)
    ~AVLNODE ()

AVLTREE

- class AVLNode
- root : AVLNode*
- insert (T, AVLNode*) : void
- insert2 (T, AVLNode*) : void
- print4 (ostream &, AVLNode*) : void
- print 3 (ostream &, AVLNode*) : void
- print 2 (ostream &, AVLNode*) : void
- print (ostream &, AVLNode*) : void
- return Object (int &, AVLNode*) : T&

+ AVLTree ()
+ find (const T &, AVLNode*) : T&
+ return Object (int &) : T&
+ ~~int~~ height (AVLNode*) : int
+ insert (const T &) : void
+ insert2 (const T &) : void
+ isEmpty () const : bool
+ rotate With Left Child (AVLNode*) : void
+ rotate With Right Child (AVLNode*) : void
+ double With Left Child (AVLNode* &) : void
+ double With Right Child (AVLNode* &) : void
+ print4 (ostream &) : void
+ print 3 (ostream &) : void
+ print2 (ostream &) : void
+ print (ostream &) : void
+ search (T&, AVLNode*) : bool
+ ~AVLTree()
+ get Root () : AVLNode*

DocumentParser

- word AVL : AVL2
- table : HashTable
- pages : AVLTree <Page*>

+ Document Parser()
+ ~Document Parser()
+ getPage AVL() : AVLTree <Page*>
+ getTable() : HashTable
+ getWord AVL() : AVL2
+ checkFor Word AVL(string &) : bool
+ checkFor Word Hash(string &) : bool
+ getInput AVL() : void
+ getInputHash() : void
+ stripUnicode (string &) : void
+ struct InvalidChar

HashTable

- count : int
- trees : AVL2*

+ HashTable()
~~+HashT~~
+ ~HashTable()
+ addWord (Word*) : void
+ getRoot (string) : AVLNODE
+ hashKey (const char*) : unsigned
+ printTrees (ostream &) : void
+ returnWord (string &) : Word*
+ searchFor (string &, AVLNODE*) : bool
+ searchTrees (string) : Word*

Page

- id : int
- title : string
- text : string

+ Page()
+ Page(string, int)
+ Page(string, int, string)
+ ~Page()
+ getId() : int
+ getTitle() : string
+ getText() : string
+ print(ostream &) : void

Index Handler

~~astIndex~~ :
+ IndexHandler()
+ ~IndexHandler()
+ print() : void

Query

- tree : AVL2
- page Index : AVLTree <Page*>
- table : HashTable
- topPage Index : AVLTree <Page*>
- searchWords : vector <string>

+ Query()
+ ~Query()
+ buildIndex() : void
+ startQuery() : void
+ frequency Sort (vector<int> &, vector<int> &) : voi
+ quickSort (vector<int> &, vector<int> &) : void
+ qAND (vector<int>, vector<int>) : vector <int>
+ qOR (vector<int>, vector<int>) : vector <int>
+ qNOT (vector<int>, vector<int>) : vector <int>

Word

- word : string
- pageId : int
- numofPages : int
- pages : AVLTree<int>
- info : map <int, int>
- vPages vector <int>

+ addPages (int pageId) : void
+ addToMap (int &, int &) : void
+ getInfo() : map <int, int>
+ getNumPages() const : int
+ getPages() : vector <int>
+ getPageId() const : int
+ getWord() const : string
+ increaseFrequency (int &) : void
+ lookForPage (int &) : bool
+ Word()
+ Word(string &)
+ ~Word()

StopWord

- stWordArray : vector <string>

+ StopWord ()
+ ~StopWord()
+ createArray(): void
+ is~~StopWord(~~ [illegible] ~~const~~
+ isStopWord(const string&): bool

User Interface

- input : string

+ UserInterface()
+ ~UserInterface()
+ startProgram () : void
+ clearIndex(): void